Friends of Oracle and Java

AMIS
Edisonbaan 15
Postbus 24
3430 AA Nieuwegein
T +31(0) 30 601 60 00
E info@amis.nl
I amis.nl
BTW nummer NL8117.70.400.B69
KvK nummer 30114159
Statutair gevestigd te Enschede

Continuous delivery Maturity model

Op gestructureerde wijze op weg naar Continuous Delivery

Auteurs
Robbrecht van Amerongen

Versienummer / status
Versie 3

Aanmaakdatum
Januari 2015

Bestandsnaam
continuous delivery maturity model v3.docx

Inhoudsopgave

1 Continuous delivery Maturity model

De methoden en technieken voor Continuous Delivery winnen steeds meer aan belangstelling. Deze aanpak blijkt vaak de succesvolle strategie achter het realiseren van echte “business agility”. Veel organisaties weten heel goed waarom Continuous Delivery belangrijk is, maar worstelen met de vraag “hoe doe je dit dan?”. Hoe start je met Continuous Delivery en hoe zorg je voor een blijvend resultaat? Het Continuous Delivery Maturity Model helpt bij het aanbrengen van structuur en verkrijgen van begrip in de kernaspecten van het invoeren van Continuous Delivery in uw organisatie.

1.1 Waarom een Maturity model?

Continuous Delivery gaat om het verkrijgen van het overzicht van alle aspecten die betrokken zijn bij het ontwikkelen en releasen van software. Voor organisaties van enige omvang is dit een proces met een groot aantal stappen en activiteiten. Het volledige proces van het ontwikkelen en releasen van software is vaak een langdurig en complex proces waarbij een groot aantal experts en afdelingen zijn betrokken die moeten samenwerken om een groot aantal obstakels te overwinnen. Dit kan leiden tot een onbeheersbare hoeveelheid activiteiten die nodig zijn voor de invoering. Veelgehoorde vragen zijn: waar moeten we starten, moeten we alles doen of kunnen we zaken achterwegen laten en welke zaken geven het snelste resultaat?
Het Continuous Delivery Maturity model geeft antwoord op deze vragen. Het biedt houvast en geeft structuur aan de invoering van Continuous Delivery en de onderliggende componenten. Het model is ontwikkeld door Andreas Rehn, Tobias Palmborg en Patrik Boström en is gebaseerd op een groot aantal artikelen en blogs over dit onderwerp, het boek “Continuous Delivery” van Jez Humble & David Farley, de whitepaper Enterprise Continuous Delivery Model en een schat aan persoonlijke ervaring. Dit model richt zich op een bredere invoering van continuous delivery die verder gaat dan “automation” en alle aspecten belicht die nodig zijn voor het invoeren van Continuous Delivery in iedere organisatie.

1.2 Een gestructureerde aanpak

Continuous Delivery is afkomstig vanuit de Agile beweging die in de afgelopen 10 jaar steeds sterker de werkwijze binnen de software-industrie bepaalt. Managers onderkennen en omarmen deze nieuwe werkwijze van softwareontwikkeling. Dit heeft tot gevolg dat er een sterke scheiding ontstaat tussen bedrijven die het niet meer lukt om de transitie te maken, en bedrijven die wel in staat zijn om wendbaar te blijven en continu één stap voor te blijven lopen op de concurrentie.
IT bedrijven die snel en effectief op veranderingen kunnen reageren, bereiken sneller een concurrentievoordeel. Deze bedrijven gebruiken IT om verder te innoveren en worden niet beperkt door kostbare, langzame en onvoorspelbare processen. Er zijn verschillende manieren om dit nieuwe tijdperk te bereiken. Dit document geeft een gestructureerde aanpak om deze resultaten te bereiken. De adoptie van veel agile methodieken groeit vaak organisch binnen een organisatie. Dit geeft in de praktijk echter een aantal beperkingen. Verschillende onderdelen van de organisatie ontwikkelen zich niet in hetzelfde tempo waardoor er verschillen ontstaan in de volwassenheid van de adoptie van Continuous Delivery. Dit heeft tot gevolg dat de invoering van dit proces beperkt wordt en dat er binnen de organisatie grotere obstakels en grenzen ontstaan.
Om deze organisatieverandering te bereiken is het realiseren van een gemeenschappelijk platform met de voorwaarden voor Continuous Delivery noodzakelijk. Dit platform zorgt voor een gemeenschappelijke roadmap en waardoor iedereen in de juiste richting beweegt. Het platform bevat de invoering van specifieke tools, principes, methoden en werkwijzen. Deze zijn gegroepeerd in vijf categorieën: Cultuur & Organisatie, Ontwerp & Architectuur, Build & Deploy, Test & Verificatie en Informatie & Rapportage.
Door Continuous Delivery in deze categorieën in te delen is het mogelijk om op een natuurlijke wijze de fasen van volwassenheid te volgen. Ook geeft deze indeling een solide basis voor een blijvend resultaat. Het doel van het model is het benadrukken van deze vijf categorieën en daarmee het verkrijgen van inzicht in de status van volwassenheid van Continuous Delivery in uw organisatie. Het model geeft een goede basis voor het invoeren

van Continuous Delivery en het helpt met het identificeren van de acties die op korte termijn het meeste effect geven. Het model geeft aan welke werkwijzen essentieel zijn voor de invoering en welke stappen er nodig zijn om naar het volgende niveau te groeien.

1.3 Continuous Delivery niveaus

Het model geeft vijf niveaus van Maturity: Base, Beginner, Intermediate, Advanced en Expert. Het model is afgestemd op de gemiddelden die we op dit moment in organisaties in de software-industrie tegenkomen. Sommige organisaties starten bij Base of zelfs daaronder (buiten het model), maar voor veel bedrijven is het Basis-niveau een goed startpunt. Het Intermediate-niveau veronderstelt een bepaalde mate van volwassenheid in Continuous Delivery waarbij een aantal van de voordelen al duidelijk zichtbaar zijn. Het Advanced-niveau brengt duidelijke en grootschalige voordelen met zich mee. Het Expert-niveau geeft aan dat de werkwijze buitengewoon waardevol is voor de organisatie en dat er behoefte is aan verdere specialisatie binnen enkelen van de methodes en technieken.
Het is niet verplicht om alle niveaus achtereenvolgens te doorlopen; het model vormt eerder een basis om de activiteiten rondom het invoeren van Continuous Delivery verder te plannen. Wel is het belangrijk om het niveau van volwassenheid (Maturity) over alle disciplines/categorieën enigszins gelijk op te laten lopen. Te grote verschillen kunnen leiden tot knelpunten en kritiek rondom de invoering in de hele organisatie. Het is aan te bevelen om de invoering stapsgewijs te laten plaatsvinden.

2 Vijf categorieën

Naast de niveaus bevat het model ook vijf categorieën die de essentiële aspecten voor de invoering van Continuous Delivery bevatten. Iedere categorie onderkent zijn eigen specifieke uitdagingen in de ontwikkeling naar volwassenheid in Continuous Delivery. Het is echter aan te bevelen om de ontwikkeling gelijkmatig over alle categorieën te laten plaatsvinden. Een ongelijke ontwikkeling in een of twee van deze categorieën kan de totale invoering negatief beïnvloeden.

2.1 Cultuur en Organisatie

De organisatie en cultuur zijn vermoedelijk de belangrijkste factoren voor het creëren van een effectieve en blijvende Continuous Delivery omgeving.

Base

Een organisatie die zich in het basis-niveau bevindt werkt volgens een geprioriteerde backlog en heeft een beperkt aantal Continuous Delivery processen die beperkt zijn gedocumenteerd. De ontwikkelteams hanteren een versiebeheersysteem waarin ze regelmatig hun sourcecode committen.

Beginner

Op het Beginner-niveau zal een organisatie teams krijgen die over verschillende afdelingen gaan. Er wordt gestart met het verwijderen van de grenzen tussen afdelingen, vooral tussen de afdelingen test en ontwikkeling. De meerdere backlogs worden geconsolideerd tot een per team. Er wordt een verdere slag gemaakt met het invoeren en adoptie van agile methodieken. Dit heeft tot gevolg dat het team verantwoordelijkheid voelt op het moment dat er zaken mis gaan.

Intermediate

Op het Intermediate-niveau realiseert de organisatie een uitgebreidere samenwerking binnen teams waarbij DBA's, Configuratiemanagers en beheer meer onderdeel van het team worden. Deze rollen worden vaker betrokken bij de werkzaamheden van het team. De verschillende processen van softwareontwikkeling worden verder geconsolideerd met als gevolg dat alle nieuwe functionaliteit, wijzigingen, bugs, hotfixes, etc. dezelfde route volgen naar productie. Ontwerpbeslissingen worden gedecentraliseerd en eigendom van componenten belegd bij de teams. Hierdoor kunnen teams zich richten op het verbeteren van de kwaliteit en stabiliteit van het component en de verbetering van het proces.

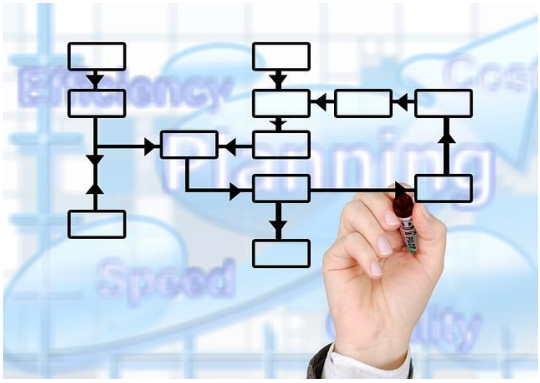
Advanced

Op het Advanced-niveau hebben de teams de kennis en het zelfvertrouwen om volledig verantwoordelijk te zijn voor alle wijzigingen tot aan productie. Continuous Improvement wordt toegepast en er ontstaan specialistische "tools"-teams die andere teams helpen bij het verbeteren van het gebruik van hulpmiddelen en automation. Op dit niveau wordt het opleveren van functionaliteit losgekoppeld van de daadwerkelijke deployment. Hierdoor krijgen de projecten een iets andere rol. Een project kan zich nu richten op het produceren van functionaliteit voor een of meerdere teams. Wanneer er voldoende functionaliteit is getest en opgeleverd naar productie, kan middels een systeem-setting deze functionaliteit geactiveerd worden. Vanaf dat moment kan een project zich richten op de daadwerkelijke oplevering naar de eindgebruikers. Deze werkwijze heeft als voordeel dat de projecten de flexibiliteit hebben om hun functionaliteit in pakketten te kunnen opleveren, onafhankelijk van de technische releases. En het geeft de ontwikkelteams een betere grip op de voortgang en productiesnelheid omdat ze geen wijzigingen op hoeven te sparen voor de project-release.

Expert

Op het Expert-niveau vormen sommige organisaties volledige cross-functionele teams die volledig autonoom kunnen opereren. Met extreem korte overtijden en een volwassen delivery – pipeline krijgt een dergelijke organisatie veel vertrouwen in het proces. Deze organisaties kunnen zich veroorloven om een “strict roll-foward” strategie te hanteren, ook bij fouten bij de productiegang. Het produceren van een fix is in deze gevallen vaak sneller dan een roll-back naar de voorgaande situatie.

2.2 Ontwerp en Architectuur



Het ontwerp en de architectuur van de softwareproducten is essentieel voor het vermogen om Continuous Delivery in te voeren. Als een product vanaf het begin is ontworpen met de principes van snel en hoog frequent opleveren, is de stap naar Continuous Delivery een stuk eenvoudiger. Voor veel bedrijven is een compleet herontwerp van bestaande systemen geen aantrekkelijke optie. Daarom is Ontwerp en Architectuur als losse categorie in het Maturity model opgenomen.

Base

Een typische organisatie op Base-niveau heeft een of meerdere monolithische legacy-systemen met bijbehorende ontwikkel-, bouw- en releaseprocessen. Veel organisaties in dit niveau beschikken over een divers applicatielandschap en zijn gestart met het consolideren van technieken en platforms. Consolidatie zal op dit niveau het grootste voordeel opleveren van de investeringen in automation.

Beginner

Op het Beginner-niveau is het monolithische karakter van applicaties aangepakt door deze op te splitsen in kleinere systemen en modules. Deze modules geven een betere structuur aan de inspanning in ontwikkeling, bouw en deployment maar de modules worden nog altijd gezamenlijk naar productie gebracht. In deze fase zal er meer aandacht zijn voor het ontwerpen en beheren van interfaces tussen verschillende systemen (API). Deze structuur zal leiden tot een betere beschrijving van de interne afhankelijkheden binnen een applicatie en een betere beheersing van het gebruik van externe libraries. Op dit niveau zal er een sterkere behoefte ontstaan aan versiebeheer. Niet alleen op de applicatiecode maar ook op de databasewijzigingen.

Intermediate

Op dit niveau zal er een solide Continuous Delivery architectuur gerealiseerd worden. Door het invoeren van abstractie (branch by abstraction) en andere technieken blijft de codebase relatief eenvoudig en wordt het onnodig branchen van de software in de repository voorkomen. Hierdoor wordt het systeem voorbereid op de daadwerkelijke stap naar Continuous Delivery. Generieke onderdelen worden ondergebracht in aparte modules die zelfstandig ontwikkeld en gedeployed kunnen worden. In deze fase worden ook de ad-hoc deployment management applicaties ondergebracht in versiebeheer. Deze krijgen vanaf dit moment dezelfde aandacht als de rest van de applicatiecode.

Advanced

Op het Advanced-niveau is het systeem opgedeeld in kleine autonome componenten die alleen via een heldere API interface met elkaar communiceren. De componenten zijn zelfstandige eenheden en op deze wijze ook onafhankelijk van elkaar te releasen en te deployen.
In deze volwassen component-gebaseerde architectuur, waar ieder component een autonome business value vertegenwoordigt, is het mogelijk om kleine en hoogfrequente opleveringen te doen met een korte release-cyclus. Op dit niveau is het voor de business eenvoudig om snel te experimenteren met nieuwe functionaliteit en via monitoring het effect te meten. Op dit niveau worden statistieken over het gebruik van de systemen belangrijker en een integraal onderdeel van het applicatieontwerp en de architectuur.

Expert

In het Expert-niveau ontwikkelt de component gebaseerde architectuur zich verder. Afhankelijkheid van gedeelde infrastructuur wordt gereduceerd. De infrastructuur wordt steeds sterker aan de componenten verbonden en gedefinieerd als “infrastructure as code”. Het resultaat is dat een component volledig vanuit het versiebeheersysteem gereproduceerd kan worden, vanaf het operatingsysteem tot aan de applicatie-interface. Deze werkwijze bespaart veel kosten en complexiteit die normaal gesproken gebruikt wordt in het zeker stellen van het kunnen reproduceren van de productie omgeving. Er is geen apart proces meer nodig voor disaster recovery. Er wordt simpelweg een nieuwe release uitgebracht vanuit de delivery pipeline, net als elke andere release. Samen met virtualisatie creëert deze aanpak een extreme flexibiliteit in het opzetten van test- en productieomgevingen met een minimum aan handmatige werkzaamheden.

2.3 Build & Deploy

Build and deployment is uiteraard de kern van Continuous Delivery en de plek waar een veelvoud aan tools en automatisering toepassing vinden in de deployment pipeline. Dit is vaak het punt waar het eerst aan gedacht wordt als het gaat om Continuous Delivery. Op het eerste gezicht kan een volwassen delivery pipeline erg overweldigend overkomen. De complexiteit van de pipeline is sterk afhankelijk van de mate van de volwassenheid van het huidige build en deployment proces van de organisatie. Deze categorie beschrijft een logisch groeimodel waarmee structuur en begrip voor de verschillende niveaus worden mee genomen.

Base

In het Base-niveau is de code opgenomen in een versiebeheersysteem en start er regelmatig een gescript build op een specifieke build-server. De deployment van software gebeurt handmatig of is in beperkte mate geautomatiseerd. Het deployment proces is beperkt gedocumenteerd.

Beginner

In het Beginner-niveau vinden de builds frequenter plaats. Hierdoor krijgt het team sneller feedback over de kwaliteit van de artefacten. De build-artefacten worden gearchiveerd en opgenomen in een systeem van dependency management. Het “taggen” en “Versioneren” van de build gaat gestructureerd, maar nog wel handmatig. Het deployment proces verloopt meer gestandaardiseerd met meer documentatie en ondersteuning door scripts en tools.

Intermediate

Het Intermediate-niveau gaat uit van builds die worden geïnitieerd op basis van wijzigingen in het versiebeheersysteem als gevolg van commits van ontwikkelaars. “Tagging” en “Versioning” gaat automatisch en het deployment proces is gestandaardiseerd voor alle omgevingen in de delivery pipeline. De deployables (build-artefacten) worden geproduceerd als release packages en slechts een keer samengesteld. Deze deployables zijn zo ontworpen dat ze op iedere omgeving geïnstalleerd kunnen worden. Het deployment proces bevat ook een basis voor automatische database deployments, data migratie, bulk database wijzigingen en gescripte configuratie wijzigingen. Dit is het moment waarop een basale delivery pipeline ontstaat vanaf het versiebeersysteem tot aan productie.

Advanced

Op dit niveau ontstaat de behoefte om de build te schalen naar meerdere machines en op parallelle wijze gelijktijdig meerdere omgevingen te bedienen. De zero-downtime-deploys techniek is in deze fase belangrijk om aan de automatische processen toe te voegen. Dit geeft een hogere flexibiliteit en reduceert het risico en kosten bij een release. In dit niveau kan het verder perfectioneren van databasemigraties en het automatisch updaten van complexe databaseroutines erg waardevol zijn. Hiermee kunnen complexe en kostbare, handmatige database updates voorkomen worden.

Expert

Op het Expert-niveau realiseert de organisatie zero-touch deployments waarbij iedere commit van een ontwikkelaar potentieel volledig geautomatiseerd tot aan productie kan worden doorgevoerd. Ook zal in deze fase de infrastructuur steeds meer als code gerealiseerd worden. Met behulp van virtualisatie wordt niet alleen de

deployment artefacts, maar ook volledige virtual machines gepromoveerd door de deployment pipeline tot aan productie waar ze de bestaande machines vervangen.

2.4 Test & Verification

Testen is zonder twijfel het belangrijkste onderdeel van softwareontwikkeling en een essentieel onderdeel voor de succesvolle implementatie van Continuous Delivery. Vergelijkbaar met Build & Deploy zal de ontwikkeling van testen een grote hoeveelheid tools en automation met zich mee brengen. Ook is het belangrijk om een continu groeiend aantal tests te creëren, waardoor een hogere dekkingsgraad verkregen wordt. Het groeiend aantal tests verhoogt het vertrouwen in het werken met een snelle hoog frequente release cyclus. Traditioneel gezien valideren tests functionaliteit tegen requirements. In het invoeren van continuous delivery is het ook van belang om de verwachte “business value” van de opgeleverde features te valideren.

Base

In het Base-niveau zal het ontwikkelteam in veel gevallen werken met unit-tests die draaien op een of meerdere speciaal daarvoor ingerichte testomgevingen. Een aparte testafdeling voert de systeem- en integratietest uit gedurende een hinderlijk lange testperiode met een bijbehorende “code freeze”.

Beginner

In het Beginner-niveau start het team met het onderzoeken van de mogelijkheden voor het automatiseren van de bestaande handmatige integratie- en regressietest. Dit zorgt voor snellere feedback en een uitgebreidere regressietest. Het uitvoeren van een nauwkeurige test vereist dat het component geïnstalleerd en getest wordt in een omgeving die een zo goed mogelijke afspiegeling van de productie is, inclusief alle noodzakelijke afhankelijkheden.

Intermediate

Het Intermediate-niveau vergroot het bereik van automatische tests naar functioneel en betrekt daarin een groter deel van het component dan de unittest. Hierbij worden de externe afhankelijkheden naar bijvoorbeeld databases en andere systemen gesimuleerd door mock-implementaties. Deze tests zijn vooral waardevol in situaties waarin gewerkt wordt met een sterk component gebaseerde architectuur of in situaties waarin uitgebreide automatische integratietests lastig zijn te implementeren of dusdanig veel tijd in beslag nemen dat deze de frequentie blokkeren. In dit niveau wordt er langzaam maar zeker gedacht aan het automatiseren van delen van de acceptatietest. Waar integratietests zich vooral richten op een specifiek component, richten acceptatietests zich op meerdere componenten over verschillende systemen heen.

Advanced

Bij het Advanced-niveau draaien de acceptatietests volledig automatisch. Het is in dit niveau zelfs mogelijk om de acceptatietests op een gestructureerde wijze volledige te genereren vanuit de requirements door gebruik te maken van “specification by example” en “domain specific language”. Dit betekent dat er geen enkele handmatige test en validatie meer nodig is om het systeem te accepteren. In de praktijk zal in het proces nog altijd een aantal verkennende tests uitgevoerd worden, die vervolgens de input vormen voor nieuwe automatische tests ter verhoging van de testkwaliteit en de testdekkingsgraad. Het combineren van de testdekkingsgraad en het traceren van wijzigingen maakt risk-based testing een beter alternatief dan handmatig verkennend testen. Organisaties die op dit niveau zijn aangekomen kijken ook naar het automatiseren van performancetests en security scans.

Expert

In het Expert-niveau draait het om het valideren van de verwachte business waarde. In de praktijk wordt deze activiteit als onderdeel van het ontwikkel- en releaseproces vrijwel nooit uitgevoerd. Het valideren van verwachte business value van systeemwijzigingen wordt eenvoudiger en natuurlijker als de organisatie, cultuur en tooling een bepaald niveau van volwassenheid heeft bereikt. In dit niveau zijn de statistieken rondom tests en feedback snel en eenvoudig toegankelijk. De voorwaarde voor de implementatie van een nieuw stuk functionaliteit is dat deze ook de wijze waarop de business value gevalideerd kan worden moet bevatten. Hierdoor kunnen er direct relevante statistieken uit het systeem getrokken worden. De *definition of done* moet zich uitstrekken tot na de release, zodat de effecten van de toekomstige release hierin meegenomen kunnen worden.

2.5 Information & Reporting

Elk bedrijfsproces bevat of produceert specifieke rapportage-informatie; het proces van ontwikkeling en release van software is hier geen uitzondering op. De informatie waarover gerapporteerd kan worden, bevat concepten als component, requirement, versie, developer, release, omgeving etc. Dit onderdeel beschrijft het belang van de juiste verwerking van deze informatie in de context van Continuous Delivery. De informatie moet op het juiste moment consistent, relevant en toegankelijk zijn voor de juiste persoon zodat deze op de juiste wijze kan ingrijpen in het operationele Continuous Delivery proces. Naast de informatie die nodig is voor het ontwikkelen en releasen van nieuwe features, is het ook van belang dat er informatie beschikbaar is waarmee het proces zelf verbeterd kan worden.

Base

Op het Base-niveau is het in deze categorie belangrijk dat er een basale statistiek over het huidige proces beschikbaar is voor het meten en volgen van dit proces. Rapportage is op dit niveau in veel gevallen handmatig en op verzoek van een individueel persoon. Interessante statistieken op dit niveau zijn bijvoorbeeld doorlooptijd, oplevertijd, aantal releases, aantal hotfixes, aantal incidenten, hoeveelheid nieuwe features per release, het aantal fouten gevonden tijdens integratie test, etc.

Beginner

In het Beginner-niveau hebben de metingen als doel het proces beter te begrijpen en te identificeren waar verbeteringen noodzakelijk zijn. Deze meting is tevens een maatstaf ter controle of de aangebrachte verbetering effect heeft. Op dit niveau bestaat de rapportage uit statische analyse over de code en kwaliteit. De rapportages worden automatisch en met enige regelmaat gegenereerd, zodat de laatste versie altijd beschikbaar is en kan dienen om besluiten over kwaliteit en verbeteringen te faciliteren.
Intermediate
In het Intermediate-niveau ontstaat er een gemeenschappelijk model waarin de betekenis van de onderdelen van het gemeenschappelijke informatiemodel is gestandaardiseerd, inclusief de onderlinge relaties. Dit model zal in veel gevallen antwoord geven op vragen zoals: wat is een component en wat is de relatie tussen de functionaliteit en de componenten. Dit model kan helpen in het verder automatiseren van de build en deployment stappen. Het maakt de inhoud van de delivery pipeline transparant, het vergroot de traceerbaarheid en het genereert belangrijke informatie zoals releasenotes en testplannen.
De geautomatiseerde rapportage bevat tevens feedback en gegevens over belangrijke events en slaat deze op samen met gegevens over de builds. Deze rapportage bevat belangrijke informatie voor het nemen van cruciale besluiten over hoe het proces verder geoptimaliseerd moet worden.

Advanced

Op het Advanced-niveau zijn de tools en rapportagehulpmiddelen dusdanig volwassen dat deze automatisch statistieken produceren en deze op een grafische wijze beschikbaar stellen aan de hele organisatie. Op deze wijze kan iedereen op elk gewenst moment gebruik maken van real-time informatie die op dat moment voor die persoon relevant is. De rapportages bevatten naast real-time informatie ook historische trends. Naast de statische code analyse en rapportage over dekkingsgraad van de unit tests, komt er nu ook informatie over de dynamische test-coverage beschikbaar die direct afkomstig zijn van systemen die vrijwel identiek zijn aan de productie. Dit is het resultaat van de automatische integratietests.

Expert

Het Expert-niveau implementeert en verbetert de real-time informatie en stel deze beschikbaar via dynamische dasboards die via self-service modules door de gebruikers op maat gemaakt kunnen worden. Deze hulpmiddelen stellen de organisatie in staat om statistieken met elkaar te vergelijken en verbanden tussen verschillende organisatieonderdelen te ontdekken. Deze informatie geeft een beter beeld van de gehele organisatie en de aangebrachte verbeteringen. Het maakt direct de resultaten van ingevoerde veranderingen meetbaar.

3 Begin de reis met een vliegende start

Ieder bedrijf is uniek en heeft zijn eigen specifieke uitdagingen bij het veranderen van een werkwijze zoals de invoering van Continuous Delivery. Het Maturity model geeft een startpunt en de basis voor de transformatie van de organisatie naar Continuous Delivery. Na beoordeling van de organisatie ten opzichte van het model, kan er een inschatting gemaakt worden van de aanpassingen die moeten plaatsvinden in de werkwijze om de meest effectieve stap in de juiste richting te nemen. Als er stappen of werkwijzen zijn die de organisatie wil overslaan, aanpassen of uitsluiten, dan maakt het model de gevolgen van deze keuze zichtbaar. Ook is het belangrijk om een duidelijke implementatiestrategie te kiezen. Bijvoorbeeld klein beginnen en daarbij de ruimte nemen om bestaande processen per stuk te verbeteren. Ervaring leert dat een helder projectteam met een duidelijk mandaat en uitdagende doelen (bijvoorbeeld het terugbrengen van de doorlooptijd) een vliegende start maakt. Een normaal Continuous Delivery implementatie project moet beschikken over de vaardigheden en competenties die horen bij de categorie en niveau uit het Maturity model. Dit stelt de teams in staat om op het juiste moment de juiste tools, methoden en technieken te implementeren. Dit zorgt er voor dat de organisatie continue kan groeien en verbeterd kan worden.

	Base	Beginner	Intermediate	Advanced	Expert
Culture & Organization	• Geprioriteerd werk • Proces is vastgesteld en gedocumenteerd • Veelvuldige commits	• Een backlog per team • Gedeelde pijn • Stabiele teams • Adoptie van basis Agile methoden • Verwijderen van grenzen tussen development en test	• Uitgebreide samenwerking tussen teams • Eigendom van componenten • Reageren op metrics • Verwijderen van grenzen tussen Dev en Ops • Zelfde proces voor alle wijzigingen • Decentralisatie van besluitvorming	• Specialistische tools-teams • Team is volledig verantwoordelijk tot aan productie • Deployment losgekoppeld van release • Continuous improvement (Kaizen)	• Cross-functionele teams • Niet meer terug..altijd vooruit.
Design & Architecture	• Geconsolideerd platform en techniek	• Systeem organiseren in modules • API management • Library management • Versiebeheer op DB wijzigingen	• Geen (of minimaal) branching • Splits op abstractie niveau • Configuratie als code • Verbergen van features • Modules ombouwen tot componenten	• Volledige component gebaseerde architectuur • Push business metrics	• Infrastructuur als code
Build & Deploy	• Code in versiebeheer • Builds zijn gescript • Basis build scheduling (CI) • Dedicated build server • Deployment handmatig en gedocumenteerd • Beperkt aantal deployment script	• Frequenter builden • Artefacten worden opgeslagen • Handmatige taggen en versionering • Eerste stap naar gestandaardiseerde deployment	• Automatische build, na commit • Automatisch tag en versionering • Buidl once, deploy anywhere • Automatische DB Scripts ook voor bulk wijzigingen • Basis pipeline naar productie • Gescripte configuratie wijzigingen • Standaard proces voor alle omgevingen	• Zero downtime deployment • Meerdere build machines • Volledige automatische DB deploy	• Bakken van infra en deployable in één • Zero touch continuous deployment
Test & Validation	• Automatische unit tests • Aparte testomgeving	• Automatische integratietest	• Automatische component test • Enige automatische acceptatie test	• Volledige automatische acceptatie test • Automatische performance test • Automatische security test • Risk-based handmatig testen	• Validatie op verwachtingen aan business value
Information & reporting	• Basis proces monitoring • Handmatige rapportages	• Meet het proces • Statische code analyse • Geplande automatische rapporten	• Gemeenschappelijk informatiemodel • Traceerbaarheid in pipeline • Rapportage historie beschikbaar	• Grafische weergave • Dynamische analyse van testdekking • Rapportage en trend analyse	• Dynamisch grafisch dashboard • Cross silo analyse